ROBOT NAVIGATION:

Part #1 of Mini-Project:

Gal Gilat	207766304	gal.gilat@campus.technion.ac.il
Alexander Vasilyev	337740773	alksndro@campus.technion.ac.il
Ofek Nachshoni	212594527	ofekn@campus.technion.ac.il

Table of Contents:

Question 1

We computed configuration space (C-space) obstacles for a convex polygonal robot navigating around a single convex polygonal obstacle. We implemented the slice-based method described in Latombe's algorithm:

1. Defined the robot and obstacle as lists of vertices ordered counterclockwise.
2. For each of 32 regularly spaced orientation values $\theta \in [0, 2\pi)$, we:
   - Rotated the robot by θ
   - Reflected the robot about the origin
   - Computed the Minkowski sum of the reflected robot and the obstacle
   - Extracted the convex hull of the result to represent the obstacle in C-space for θ
3. Plotted the resulting polygonal C-obstacle slices in the $x - y$ plane for each θ, and printed the results for slices 1, 8, 16, and 32.

This allowed visualization of how the robot's configuration constraints change with orientation relative to the obstacle.

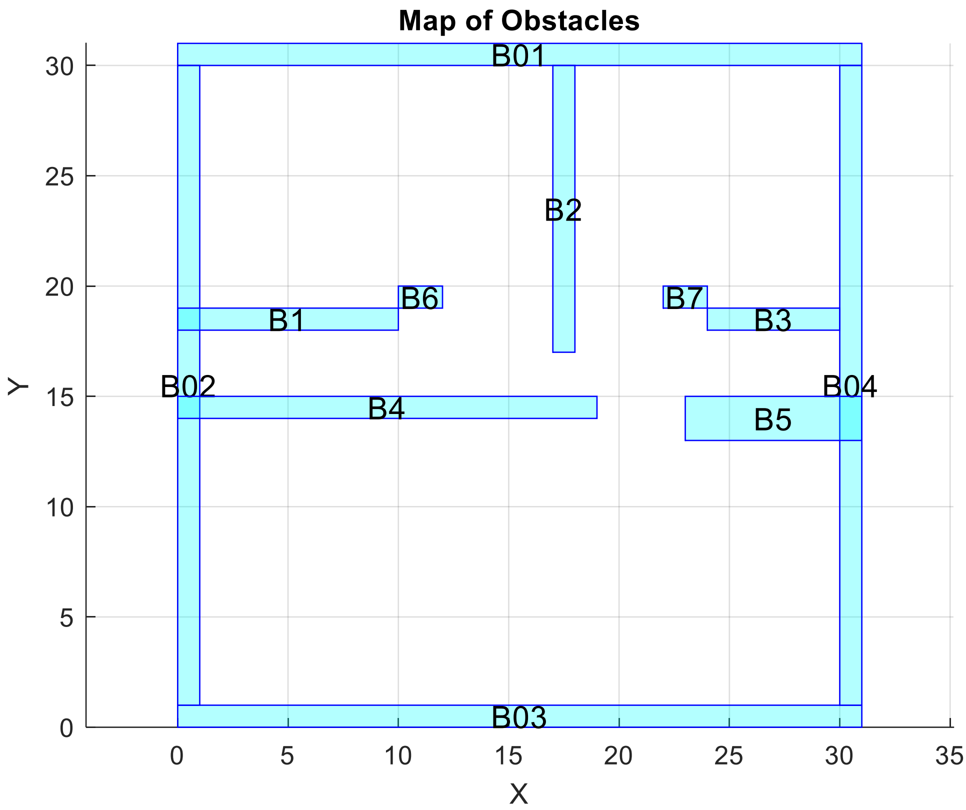

figure 1: Map of Obstacles

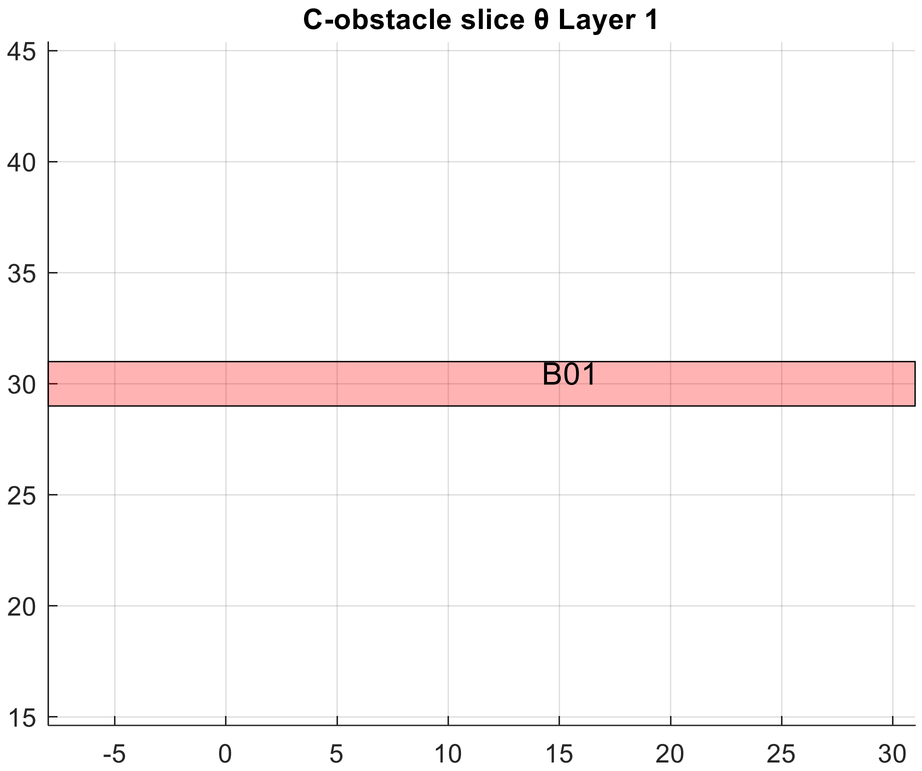

figure 2: Layer 1

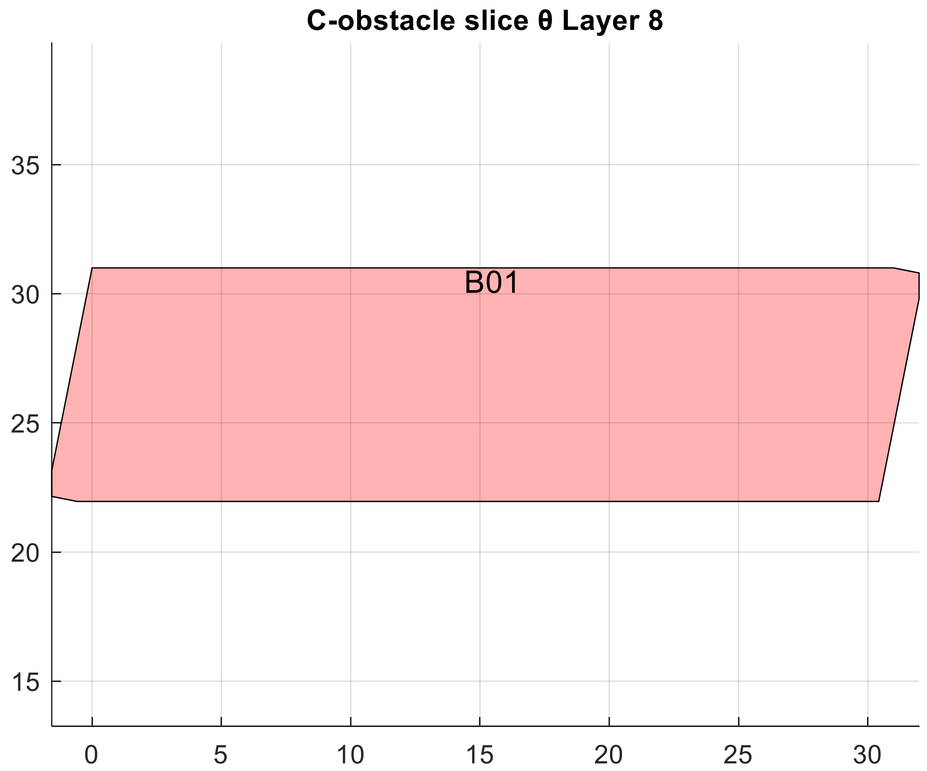

figure 3: Layer 8

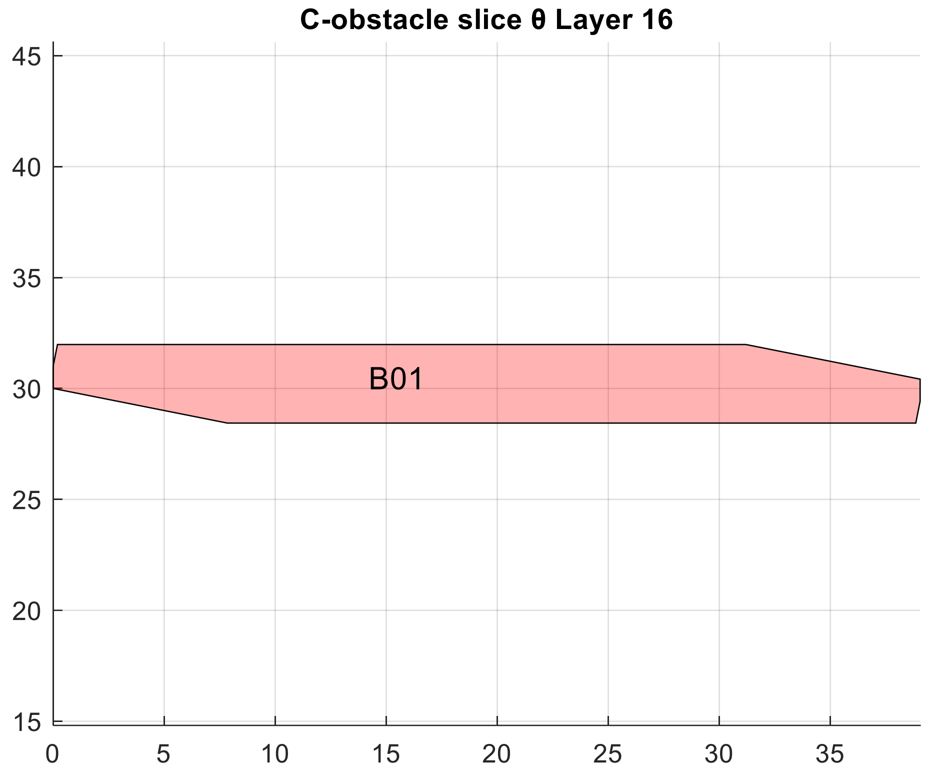

figure 4: Layer 16

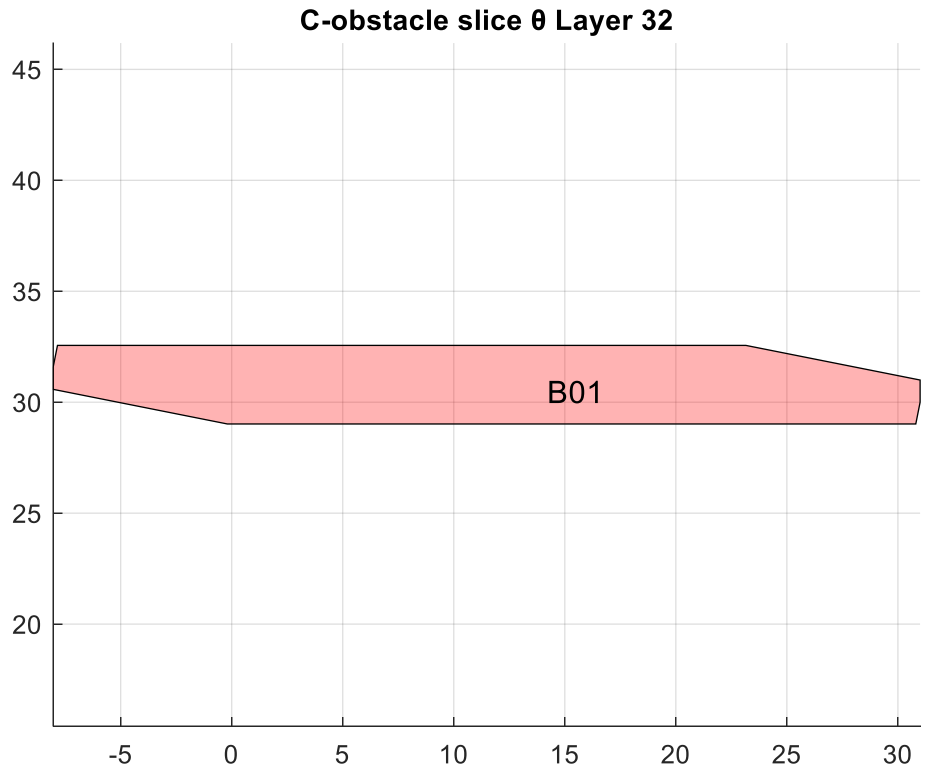

figure 5: Layer 32

Question 2

In Question 2, we extended the solution from Question 1 to handle a realistic environment with multiple convex polygonal obstacles. The environment consists of outer walls (B01–B04), internal walls (B1–B5), and doors (B6–B7). We used the same algorithm as in Question 1 to compute the configuration space (C-space) obstacle slices for each obstacle individually over 32 discrete rotation angles of the robot.

For each orientation θ, we computed the Minkowski sum of the rotated robot with every obstacle and extracted the convex hull to obtain the C-obstacle slice. We then overlaid the slices for all obstacles in a single plot for θ-layers 1, 8, 16, and 32. This provided a layered visualization of the full configuration space and demonstrated how the robot's collision constraints evolve with rotation in a complex environment.

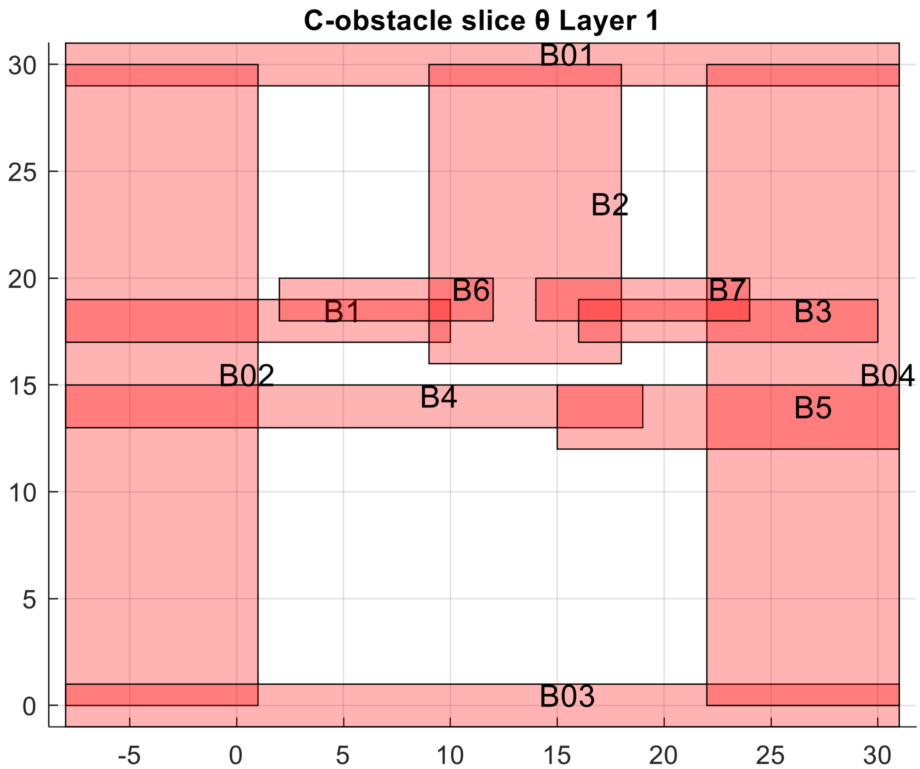

figure 6: Layer 1

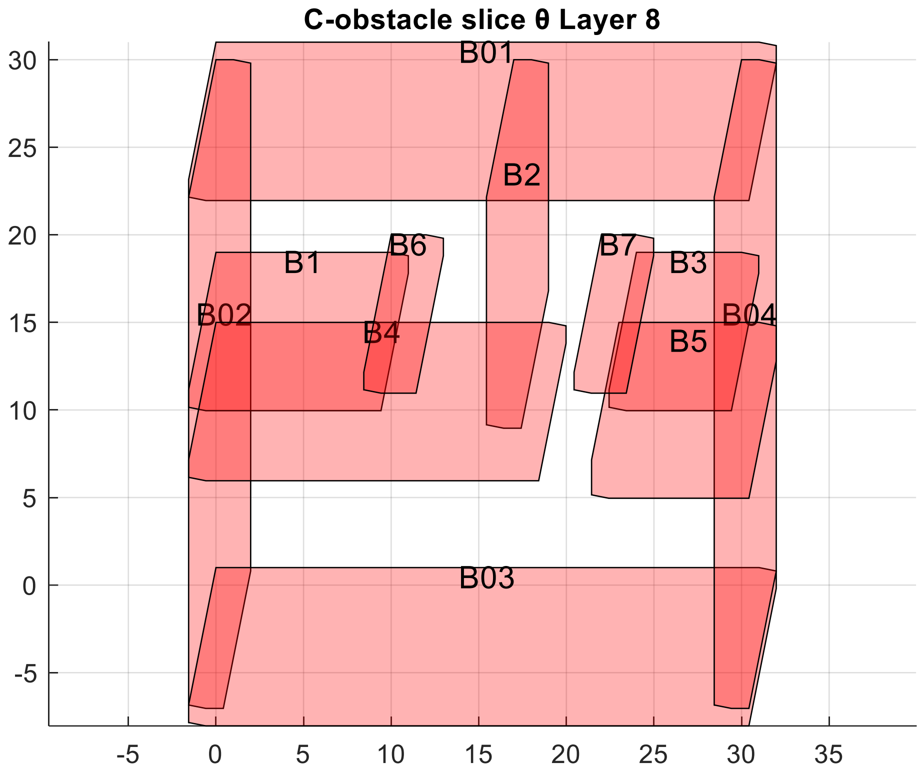

figure 7: Layer 8

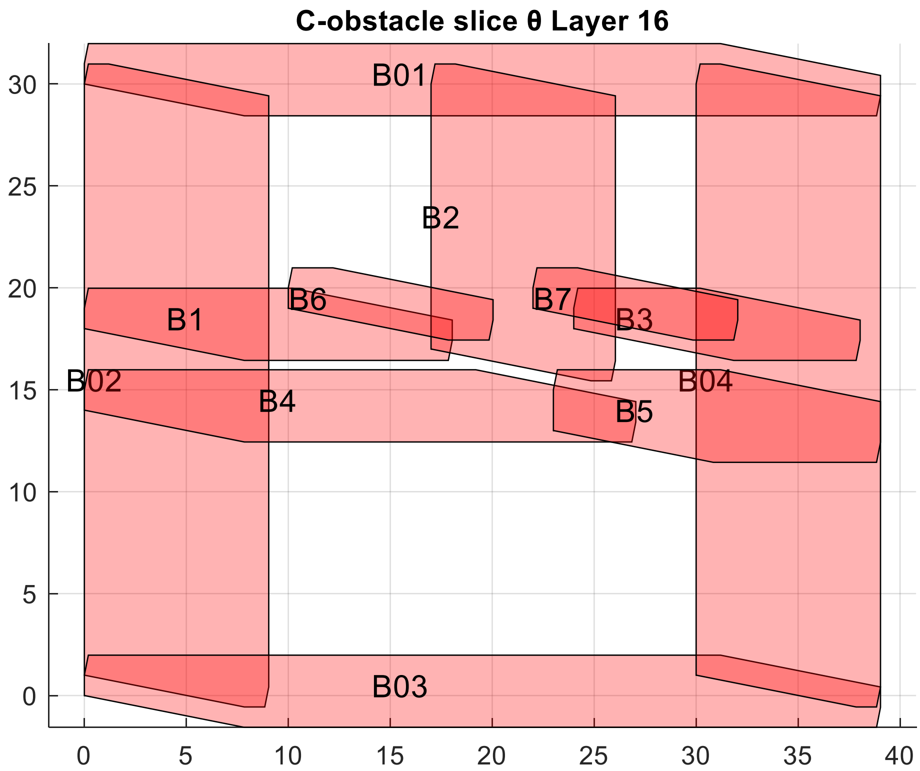

figure 8: Layer 16

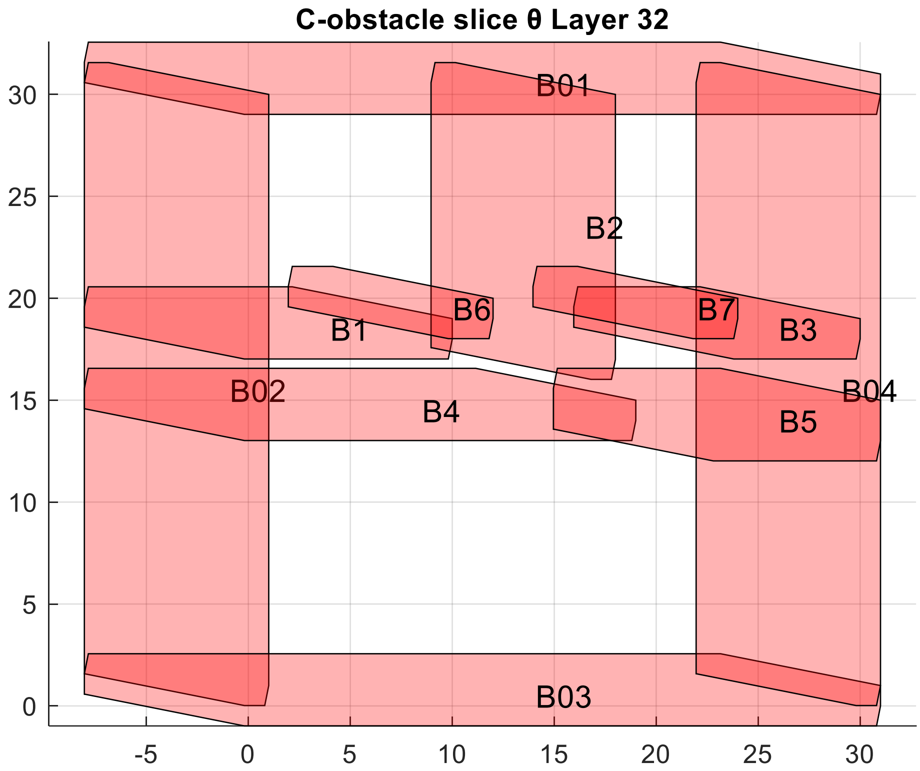

figure 9: Layer 32

Question 3

In question 3, we took the printouts of layers 1,8,16,32 and put a “1” value for each cell with the c-obstacles boundaries. All other cells were filled with “0” value. We then plotted each layer and outline the obstacles. Here is the matrices for each layer:

<u>For Layer 1:</u>

1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
1	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

For Layer 8:

0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	1	0	1	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	0	1	0	0	0	0	0	1	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	1	1	1	1	1	1	1	1	0	0	1	0	1	0	1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
1	0	1	1	0	0	0	0	1	1	1	1	1	0	1	0	1	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	1	1	0	1	1	1	0	0	1	0	0	0	1	0	1	1	0	0	0	0	0	1	0	0
0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	0	1	0	0	0	1	1	0	1	0	0	0	0	0	1	0	1
0	1	0	1	0	0	0	1	1	0	1	1	0	1	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	1	0	1
1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	0	1	0	0	0	1	1	0	1	0	1	0	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	1	0	0
1	0	0	1	0	0	0	1	0	0	1	0	1	0	0	0	1	0	0	1	1	0	1	0	0	0	0	0	0	1	0	1
1	0	0	1	0	0	1	1	0	1	0	0	1	0	0	0	1	0	1	0	1	1	0	0	0	0	0	0	0	1	0	1
1	1	1	1	1	1	1	1	0	1	0	0	1	0	0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	0	1	1	1	1	0	0	1	0	0	0	1	0	1	1	1	1	0	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

For Layer 16:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1	1
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	1
0	1	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	0	0	1
0	1	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1
0	1	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0	0	1	1	1	1	0	0	1
0	1	0	1	1	1	1	1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	1	0	0	0	0	0	0	1	1	1
1	1	0	0	0	1	0	1	1	1	0	1	1	1	1	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	0	1	0	0	0	0	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1
0	1	0	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	1
0	1	0	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1
0	1	0	0	0	0	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1
1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

For Layer 32:

1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	1	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	1	1	1	0	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	1	0	1	0	1	1	1	1	1	0	0	0	1	0	1	1	1	1	1	1	0	0	0	1	0
1	0	0	0	1	1	1	1	1	1	0	0	0	1	0	1	1	1	0	1	1	1	1	0	1	1	1	1	1	1	1	0
1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	1	0	0	0	0	1	1	1	1	0	0	1	1	1	1
1	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1
1	1	1	1	1	1	1	0	0	1	0	0	0	0	0	0	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0

Here are the grids we got for each layer:

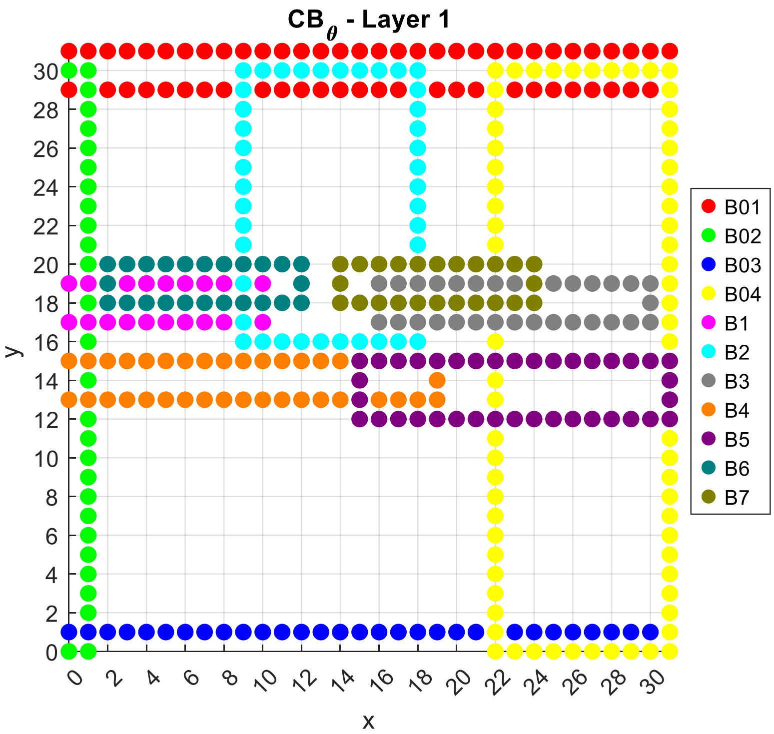

figure 10: Layer 1

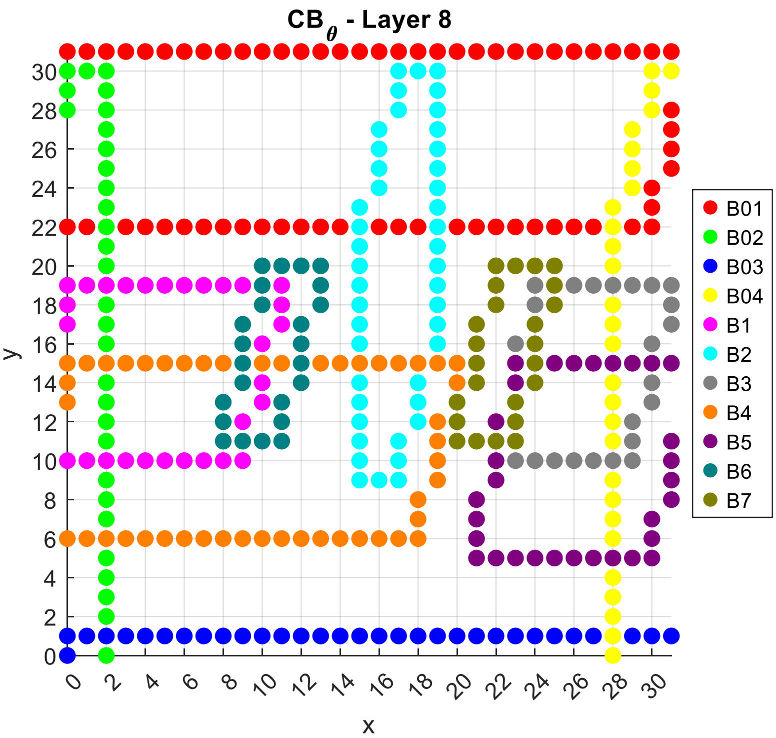

figure 11: Layer 8

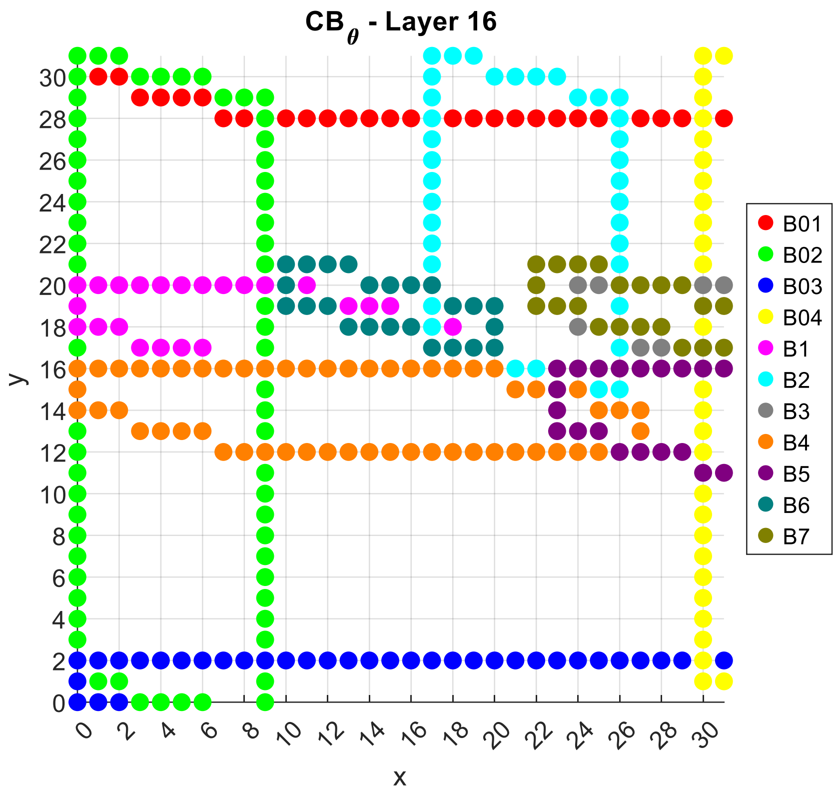

figure 12: Layer 16

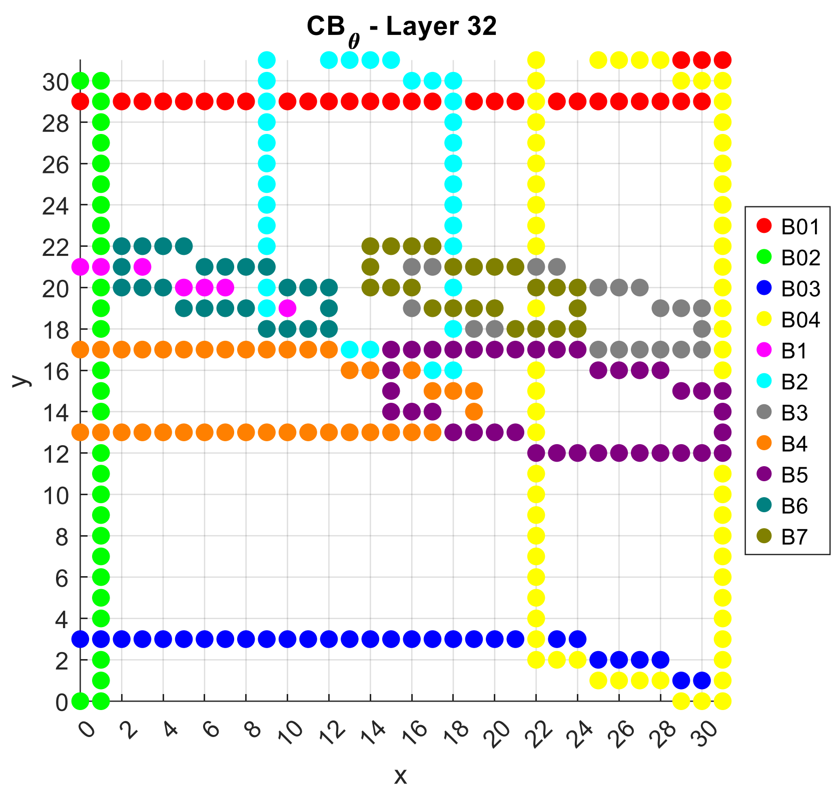

figure 13: Layer 32

After seeing those results, we thought it is a bit flawed- because it gives room for the robot to be inside the obstacles, so in our opinion it's a better approach to also fill the inside of the obstacles and not just their boundaries- thus remaining the true free space for the robot to move in. The results we got:

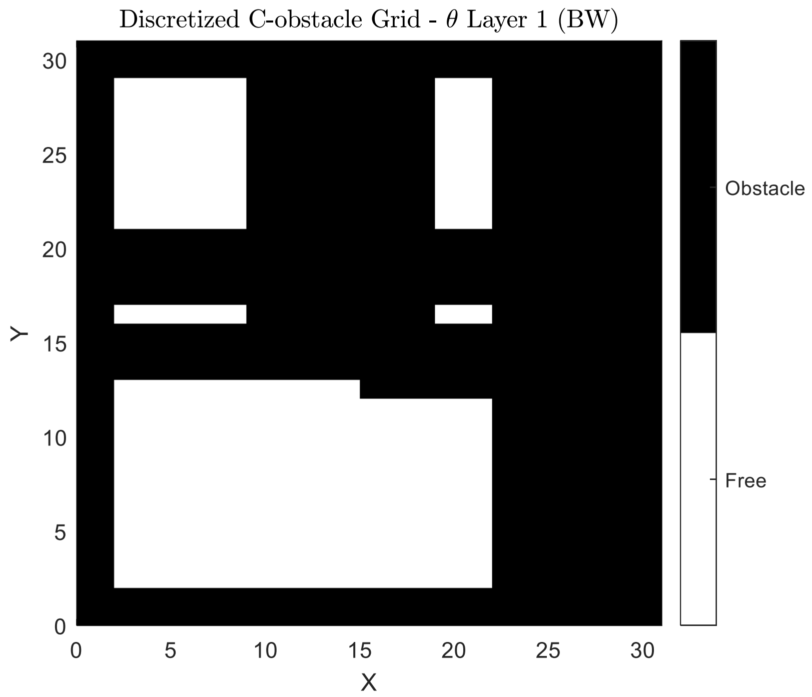

figure 14: Layer 1

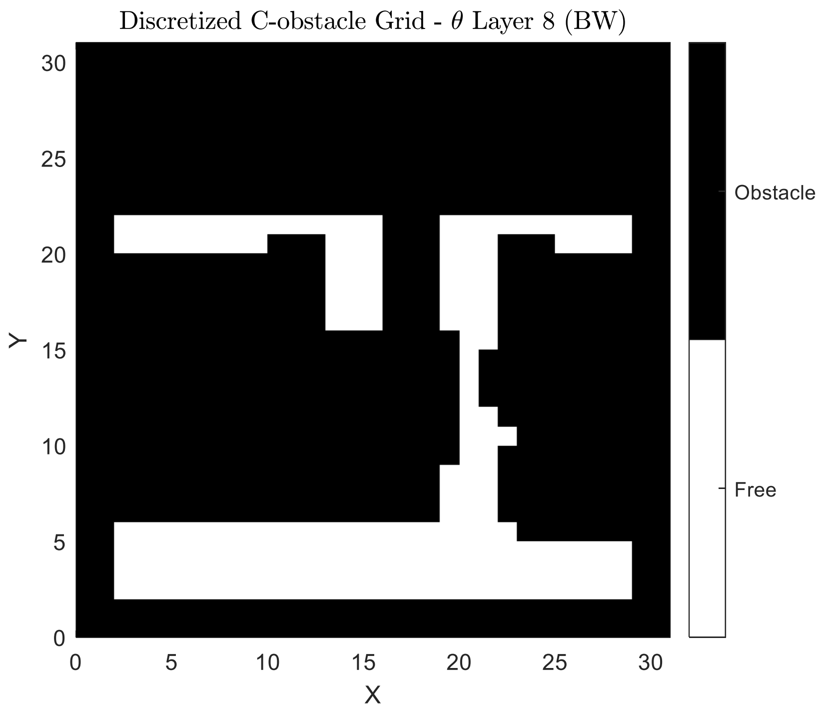

figure 15: Layer 8

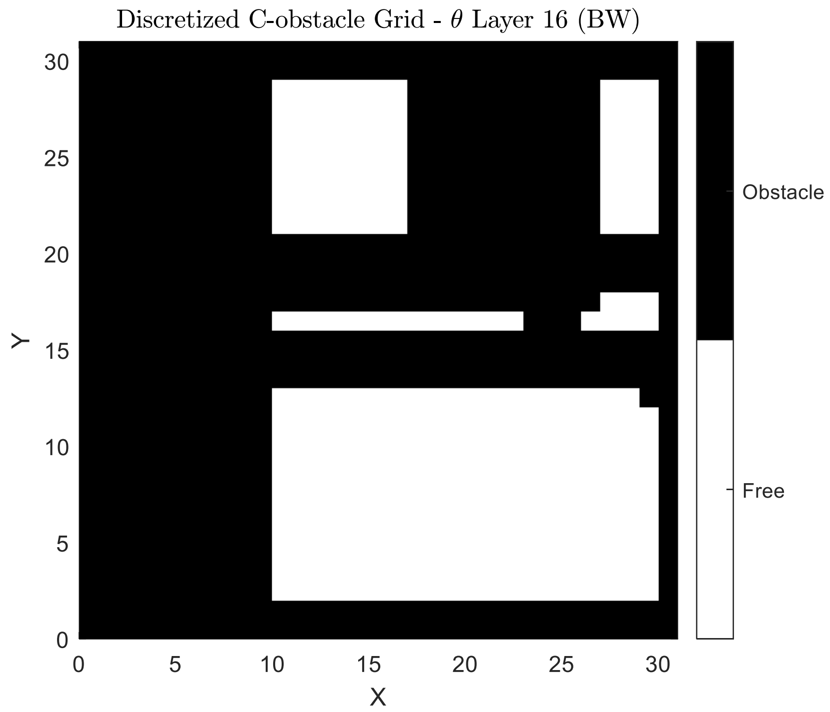

figure 16: Layer 16

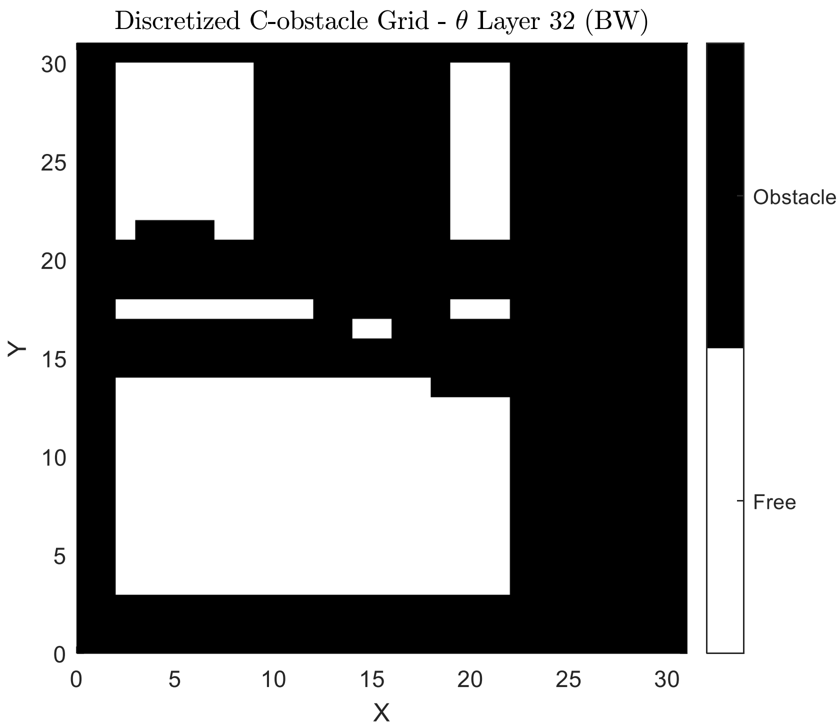

figure 17: Layer 32

The code we used in Q1 as requested:

```
close all; clear all; clc;
import myfunctions.*
%% definitions
A = [0 0; 8 0; 8 1; 0 1];
B01 = [0 30; 31 30; 31 31; 0 31];
B02 = [0 1; 1 1; 1 30; 0 30];
B03 = [0 0; 31 0; 31 1; 0 1];
B04 = [30 1; 31 1; 31 30; 30 30];
B1 = [0 18; 10 18; 10 19; 0 19];
B2 = [17 17; 18 17; 18 30; 17 30];
B3 = [24 18; 30 18; 30 19; 24 19];
B4 = [0 14; 19 14; 19 15; 0 15];
B5 = [23 13; 31 13; 31 15; 23 15];
B6 = [10 19; 12 19; 12 20; 10 20];
B7 = [22 19; 24 19; 24 20; 22 20];
B_names = {'B01', 'B02', 'B03', 'B04', 'B1', 'B2', 'B3', 'B4', 'B5', 'B6', 'B7'};
B_list = {B01, B02, B03, B04, B1, B2, B3, B4, B5, B6, B7};

figure; axis equal; grid minor; hold on;
xlabel('X'); ylabel('Y');
title('Map of Obstacles');
for idx = 1:length(B_list)
  B = B_list{idx};
  fill(B(:,1), B(:,2), 'c', 'FaceAlpha', 0.3, 'EdgeColor', 'b');
  text(mean(B(:,1)), mean(B(:,2)), B_names{idx}, 'HorizontalAlignment', 'center', 'FontSize', 12);
end
hold off;
```

```
%% Q1

obstacle_list = {B01};

obstacle_name = {'B01'};

slices = cspace_slices_multiple(A, obstacle_list);

layers_to_plot = [1, 8, 16, 32]; % Change as needed

for layer = layers_to_plot

  figure; axis equal; grid minor; hold on;

  title(sprintf('C-obstacle slice θ Layer %d', layer));

  for idx = 1:length(obstacle_list)

    obstacle = obstacle_list{idx};

    slice_points = slices{layer, idx};

    fill(slice_points(:,1), slice_points(:,2), 'r', 'FaceAlpha', 0.3);

    text(mean(obstacle(:,1)), mean(obstacle(:,2)), obstacle_name{idx}, 'HorizontalAlignment',
'center', 'FontSize', 12);

  end

  hold off;

end


function all_slices = cspace_slices_multiple(A, B_list)

  % A: Nx2 vertices of robot A

  % B_list: cell array of Mx2 obstacles

  theta_values = linspace(0, 2*pi - 2*pi/32, 32);

  all_slices = cell(32, length(B_list));

  for k = 1:32

    theta = theta_values(k);

    R = [cos(theta), -sin(theta); sin(theta), cos(theta)];

    rotated_A = (R * A')';

    for idx = 1:length(B_list)

      B = B_list{idx};

      B_col = B'; % 2xM

      diff_points = [];
```

```
        for i = 1:size(B_col,2)
            for j = 1:size(rotated_A,1)
                diff_points = [diff_points, B_col(:,i) - rotated_A(j,:)'];
            end
        end
        points = diff_points';
        K = convhull(points(:,1), points(:,2));
        all_slices{k, idx} = points(K, :);
    end
  end
end
```